Laborator 1
Deadline: saptamana 4

Se considera o imagine reprezentata printr-o matrice de pixeli, F , de dimensiune (MxN).
Se cere transformarea ei aplicand o filtrare cu o fereastra definita de multimea de indici W cu coeficientii w_{kl} (reprezentati prin matricea W[k,l], unde -n/2<=k<=n/2, -m/2<=l<=m/2; si n<N, m<M, n,m impare).
Transformarea unui pixel:

$$v(m,n) = \sum_{(k,l)\in W} w_{kl} f(m-k, n-l)$$

De exemplu:

$$W = \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \boxed{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$$

v[m,n] =
f[m,n] *1/9+
f[m-1,n]* 1/9+
f[m,n-1]* 1/9+
f[m-1,n-1]* 1/9+
f[m+1,n]* 1/9+
f[m,n+1]* 1/9+
f[m+1,n+1]* 1/9

Pentru frontiere se considera ca un element este egal cu elemental din celula vecina din matrice
f[-1,j]= f[0,j]; f[i,-1]=f[i,0]; f[M,j]= f[M-1,j]; f[i,N]=f[i,N-1];

Exemplificare -> https://de.wikipedia.org/wiki/Datei:2D_Convolution_Animation.gif

Se cere asigurarea urmatoarei postconditii:
Postconditie: Matricea rezultat V contine imaginea filtrata a imaginii initiale F (unde V != F)

A) Program secvential
B) Program paralel: folositi **p** threaduri pentru calcul.

Obiectiv: Impartire cat mai echilibrata si eficienta a calculul pe threaduri!
Pentru impartirea sarcinilor de calcul (taskuri) se foloseste descompunere geometrica care poate fi
(puteti alege o varianta sau sa incercati mai multe si sa o identificati pe cea mai buna):

- Pe orizontala (mai multe linii alocate unui thread)
- Pe verticala (mai multe coloane alocate unui thread)
- Bloc – submatrici alocate unui thread
- bazat pe o functie de distributie prin care unui index al unui thread i se distribuie o submultime de indecsi din matrice;
  distributia se poate face prin:
  - distributie liniara (indici alaturati la acelasi thread) sau
  - distributie ciclica(cu pas egal cu p).

Datele de intrare se citesc dintr-un fisier de intrare “date.txt”.
(Fisierul trebuie creat anterior prin adaugare de numere generate aleator. Toate rularile trebuie executate cu acelasi fisier.)

Implementare:

a) Java
b) C++ (cel putin C++11)
   i. matricile sunt alocate static (int f[MAX][MAX])
   ii. matricile sunt alocate dynamic (new…)

Folosire directa a threadurilor (creare explicita) => nu se permite folosirea executorilor.

Testare: masurati timpul de executie pentru

1) N=M=10 si n=m=3; p=4;
2) N=M=1000 si n=m=5; p=2,4,8,16
3) N=10 M=10000 si n=m=5; p=2,4,8,16
4) N=10000 M=10 si n=m=5; p=2,4,8,16

Rezultatele acestor teste trebuie sa fie reflectate in documentatie in tabele

Java:

Tip matrice	**Nr threads**	**Timp executie**
N=M=10 n=m=3	secvential	….
	4	….
N=M=1000 n=m=5	secvential	….
	1	….
	2	….
	4	….
	8	….
	16	….
….	….	….

C++

Tip matrice	**Tip alocare**	**Nr threads**	**Timp executie**
N=M=10 n=m=3	Static	4	….
	dinamic	4	….
N=M=1000 n=m=5	static	1	….
		2	….
		4	….
		8	….
		16	….
	dinamic	1	….
		2	….
		4	….
		8	….
		16	….
….	….	….	….

ObservatII:

- Fiecare test trebuie repetat de 10 ori si pentru evaluarea timpul de executie se considera media aritmetica a celor 10 rulari.
- Pentru fiecare varianta (secvential, paralele) folositi acelasi fisier “date.txt”;

Analiza

Comparati performanta pentru fiecare caz – secvential versus paralel si variantele paralele intre ele.
Comparati timpii de executie obtinuti cu implementarea Java versus implementarea C++.
Comparati cele doua variante pentru implementarea C++.
Analiza trebuie evidentiata in documentatie.